# Notes

1. Blend the business process to user stories – focus on describing the minimum products in your own words than showing the process/ mechanism

* Don’t use value proposition canvas in paragraphs. Use your own words in future work.

1. Evaluation metrics – completion rate, error rate…

# Project Proposal v1s

Laser Mate! is a self-defined start-up dissertation project that has the potential to hit over a £2.5 billion annual profit. The goal of the software is to provide a platform for restaurant customers to order and pay for their meals online.

The unique prospect of this new system is the fact that it suppresses traditional ordering methods by the costs needed to perform the ordering duty, resulting in a saving of £16,800 - £33,600 per year for the restaurant owners. This comes from the fact that they will not need to take, record, and deliver the orders and give and take meal’s payment as customers will perform these tasks instead. Since restaurant waiters would only require taking the customers to the table and deliver the meals, they will save at least 50% of the work. By reducing the number of people needed to maintain the restaurant waitering operations by 50%, restaurant owners would save an annual staff cost of 50%. A typical restaurant will have two waitering staff. Therefore, we could help restaurant owners save an annual cost of £8 x 6 hours x 350 days = £16,800. Furthermore, restaurant owners will save additional work on managing staff rota, training, supervision, and accounting.

The revenue-cost analysis indicates that each restaurant account signup would attract a £5,500 profit each year. Firstly, the transaction cost in mobile web is 1% lower than that in bank card (0.39% + 2p vs. 1.75% per transaction), allowing us to have an additional 1% profit. Taking 20% of the £16,800 staff-saving cost (mentioned above) as a commission and additionally the 1% platform transaction fee difference, we would earn £7,000 per year per restaurant. Consequently, it would result in a £5,500 profit after cost deductions for accounting and insurance.

In terms of the annual profit analysis, given we have 1.5 million restaurants in the E.U. and U.S., we will hit £2.5 billion at 30% market penetration (30% x 1.5m x £5,500).

Our calculation neglects the start-up cost. These include software expert costs, business registration, legal policies, terms and conditions, cookies, deployment cost, Q.R. code generators, post-marketing, and base salary. Our utmost priority is to prevent data breaching and software breakdown (due to high user traffic). To maintain the platform’s security and reliability, we are looking to partner with an experienced software engineer with a share option to take care of these software aspects.

Literature review also suggests that this project idea present a market gap as only Dines have an adequate software design and implementation with a similar focus. Other companies, such as Yo! Sushi, and Starbucks, only have this project idea for their own business operations.

# Project Overview

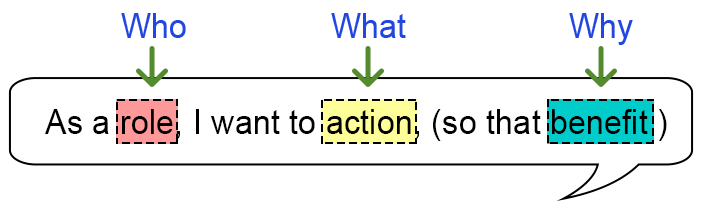
## Final Software Product v1s

## Software Requirement Statements v1s

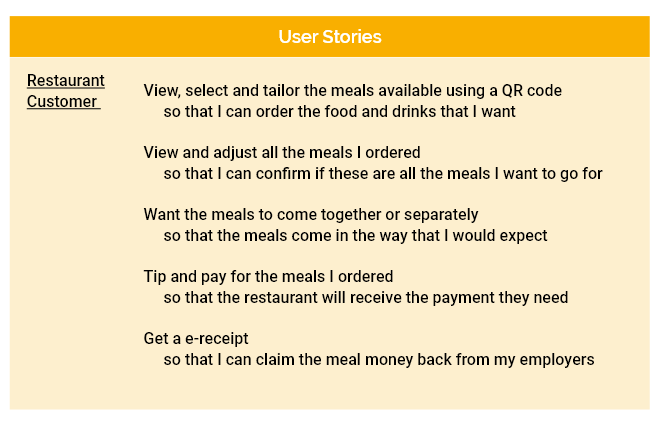
### User Stories v1s

// delete this paragraph because there is no need to explain the mechanism used to compute the software requirement?

The goal of a user story is to generically define the software’s user requirements. By having a breakdown of the main software tasks, it establishes an understanding of the possible software features. In the user stories, Role describes the system’s user groups that share a set of predicted tasks. Action(s) are all the generic tasks they will perform when using the software. Benefit justifies why they need to perform these actions.



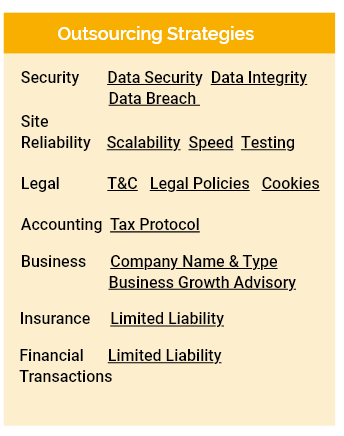
//



### X Minimal Business Executive Strategy v1s

// delete all sections and use the user stories model to describe the business workflow instead

A reputable software should consider the critical business operations as any design alternations will lead to additional time lost due to redundant software development. The minimal business executive processes, excluding customer activities, are software security and reliability; post-marketing; product review; account registration; platform setup; Q.R. codes postage; and maintenance.



Our software will follow adequate justificatory framework. We will recruit accredited security professionals to ensure that high user traffics will not compromise the platform’s speed and operations. Our outsourcing security team will protect the system from malicious attacks and data breaches.

Following system integrity, we will begin our customer acquisition process. Post-marketing is the optimal first-wave advertisement methods since stamp, envelop, paper, and printing costs are £1.5 per letter. Compared to all other marketing mediums, it is a marketing strategy that almost guarantees that all the restaurant owners would read the advertisement contents. When the restaurant owner receives the letter (Appendix Letter), they will see a Q.R. code that will redirect them to the company’s YouTube channel, demonstrating the software products. Video presentation should be the most potent persuasion mechanism to attract clients with the software capabilities and benefits.

Restaurant owners who express an interest will scan the Q.R. code and go to the website (picture). The landing page aims to demonstrate our product and the benefit of becoming a member, using minimalistic representation, such as short texts and pictures.

Perspective members could sign up by using our registration page or email. To complete this process, they would only need to fill in 11 pieces of information. We aim to do all the tasks on our client’s behalf to minimize their inputs and efforts. We will deploy a standard automatic email confirmation system to verify the customer’s email address.

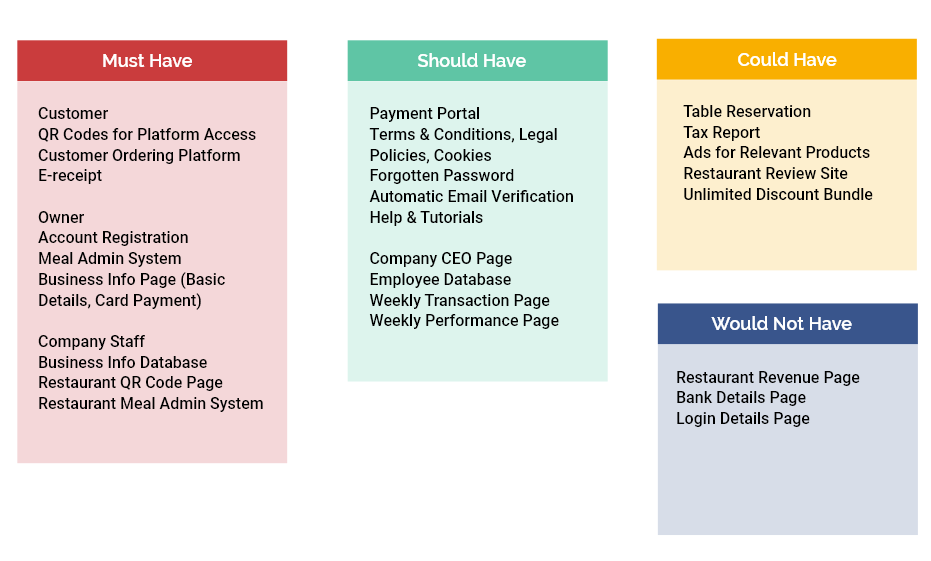
The most important aspect of the business process is to set up the restaurant sit-in ordering system with all the business info and menu. Our staff will be fully responsible for this process as the platform must have the correct presentations before deployment. Hopefully, this will prevent clients from feeling frustrated learning the system.

Following the production stage of the restaurant platform, we will print out the table Q.R. codes for later delivery. All the Q.R. codes will have lamination to protect damage due to long-term use. To ensure an appropriate number of replacements Q.R. codes, we will prepare the number of Q.R. codes three times the number of the tables.

Members with successful signups should now be able to enjoy our software platform. Hopefully, they would need minimal support and intervention as we will upload help and video tutorials to guide them with the platform’s use.

### MOSCOW v1s

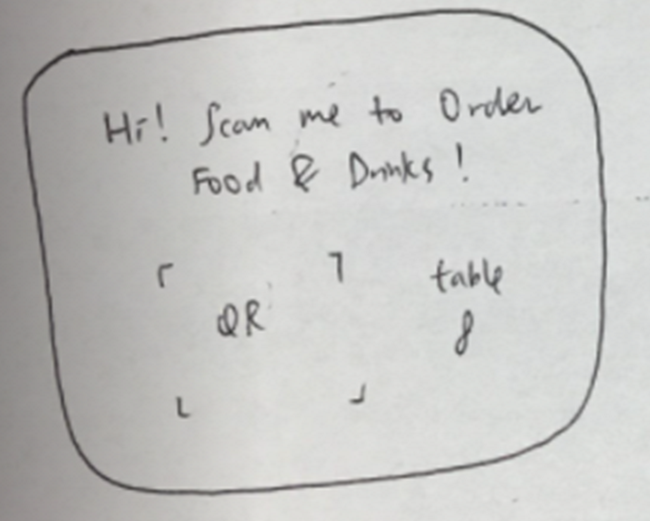
The MOSCOW method prioritizes the software features to implement. We will first develop all the “Must Have” features before the “Should Have” features. Subsequently, we will implement the “Could Have” features. Since the “Would Not Have” features are the ones not to include in the software development planning, they will be in the record, but get ignored.



## Software Design & Coding Strategies

### Paper Prototyping v1s

Paper prototyping is a software design strategy that composes all project ideas into a product. By having a drafted system in paper, it enables several first-phrase evaluations using usability heuristic evaluation and with other user groups, ensuring that the product captures all the critical business and user requirements. Another advantage of having a paper draft before putting it into computer design is that you could add and discard undesirable software components quickly by writing it up and crossing them out easily.



### 3 Justifying Design Options v1

<https://razorware.wordpress.com/2012/01/04/task-3-the-fundamental-principles-of-hci/>

https://coolors.co/palettes/trending

Prior to software development, we have refined a list of design criteria to ensure that our platform design is consistent and adhere to industrial standards. Users will only want to reuse the app if it is comfortable and engaging to use.

Inclusive Design – Understanding that many people could have difficulty using mobile devices, we design the system considering specific difficulty and conditions. We closely adhere to the colour-blind design criteria, including the use of high contrast colours, particularly in the food category, price and menu description sections. The system also adopts simple word options for people with language barriers. Another issue is eye-sight issue – all the texts are at least 16px. Secondary texts are also at least 2px smaller than the upper text sections.

Ethical Design – Our app also follows the ethical principles set out by the University, protecting the rights and wellbeing of our evaluation participants and app users. In the Appendix, you can see the ethical consent form for all our evaluations. In the evaluation section below, you can also see the ethical consent section. Furthermore, our landing page will include the company’s terms and conditions, legal policies, and cookies, establishing mutual agreement through our liability limitation statement. This section is currently fictious since our lawyer are more adequate in this domain.

Multimodal Interaction - The software would have an optional sound notification each time a customer places an order since the waiter could not possibly pay attention to incoming orders all the time.

<https://www.colourblindawareness.org/colour-blindness/>

<https://www.designmantic.com/community/website-design-guide-color-blind.php>

<https://learnui.design/blog/mobile-desktop-website-font-size-guidelines.html>

Search Engine Design – As we are dealing with over 100,000 tuples in most database interfaces, we would need to deploy a search functionality. Once the system returns a search result based on a key term, the database within our system will display all the data entries exhaustively to the end of the list, facilitating the use of CTRL+F keyword. After considering the database design with the practical system’s use, it turns out that the possible terms will not lead to crashing. The possible search terms are so different that the search results can only base its outcome on one data header. For example, in the basic restaurant info page, the possible search terms are restaurant no, restaurant name, owner name, restaurant address and telephone number. The pattern of data entries is number, text, postcode, telephone number, respectively. As a result, due to their natural text patterns, it is impossible that the complete search results will base on two different database headers. Therefore, we do not need to consider higher level search techniques, such as isolating search term to a specific header.

Scenario Documentations - One of the business protocols is to refine our help documentations. This is a record that provides case analysis and mitigation methods for all company staff, ensuring that we counter for most repetitive incidence and potential breaches. We will regularly create and update cases to increase our scenario coverage.

Internationalisation – Global reach is one of the agenda of Laser Mate!. We aim to abstract the platform so that the database will render all the texts in the platform, such as the company name, category name, and pay button. This will allow for easy text alternation into different languages, compared to creating a new platform and hand coding each word when reaching to a country whose primary language is not English.

Capitative Design Principles – Our platform design adopts a balanced consideration between software components, in terms of information, spacing, colour and font. It emphasises the importance of images over texts and minimal information representation. A design criterion we follow is information hierarchy. We place important information at an outer area of the app with a larger font size, guiding users to understand the software. We leave sufficient spacing between software components, increasing the system’s clarity and comfortability of use. Another design consideration we take is the adoption of the colour palette [?]. Coolors is a website that collects complementary colours our website designs base.

### 2 Front-End Design and Coding v1

The strategy for converting paper prototype to digital wireframe and subsequently React.JS codes is the use of several platforms. Namely, they are Adobe XD, Anima and YouTube.

The main reason to use Adobe XD, other than the common functionalities, is the new feature Anima. Given a complete digital design, it allows for direct code conversion from digital wireframes to codes within 3 minutes. Compared to traditional methods, such as Bootstrap and material designs, which could take weeks, you could say an astronautical elevation of working pace since you have a completely auto-generated codes without any human intervention. Having said that, you still need to program parts of the platform, such as those depending on the database and page interactions.

Consequently, component naming and merging becomes critical in the digital wireframe’s design, as they will be automatically converted to SCSS and JS codes. Meaningful naming in the program could help future code editing and reviews by other colleagues. Furthermore, another impressive function of Adobe XD is the “make component”, in which you can group several related items together. This will enable you to replicate the entire merged components across the platform, for example, you could merge all the components for the navigation bar and apply the same design across the meal platform. By ensuring that your designs are not slightly different (by a couple of spacing), you will not have minor code deviations within your program, enhancing code convertibility and consistency.

Another feature in Adobe XD is responsive design, in which each design components change placement according to all possible screen sizes, ranging from 320px to 414px. It controls how each software components move with respect to the four directional changes, in terms of its placement and size. A four axial icon (Figure) denotes whether the distance of the center of the icon from the side should be constant. If the distance should not change when screen resizing, it should be blue; otherwise, it should be black. The two width and height fixed icons (Figure) changes the icon’s size (in terms of width and height) when resizing the screen.

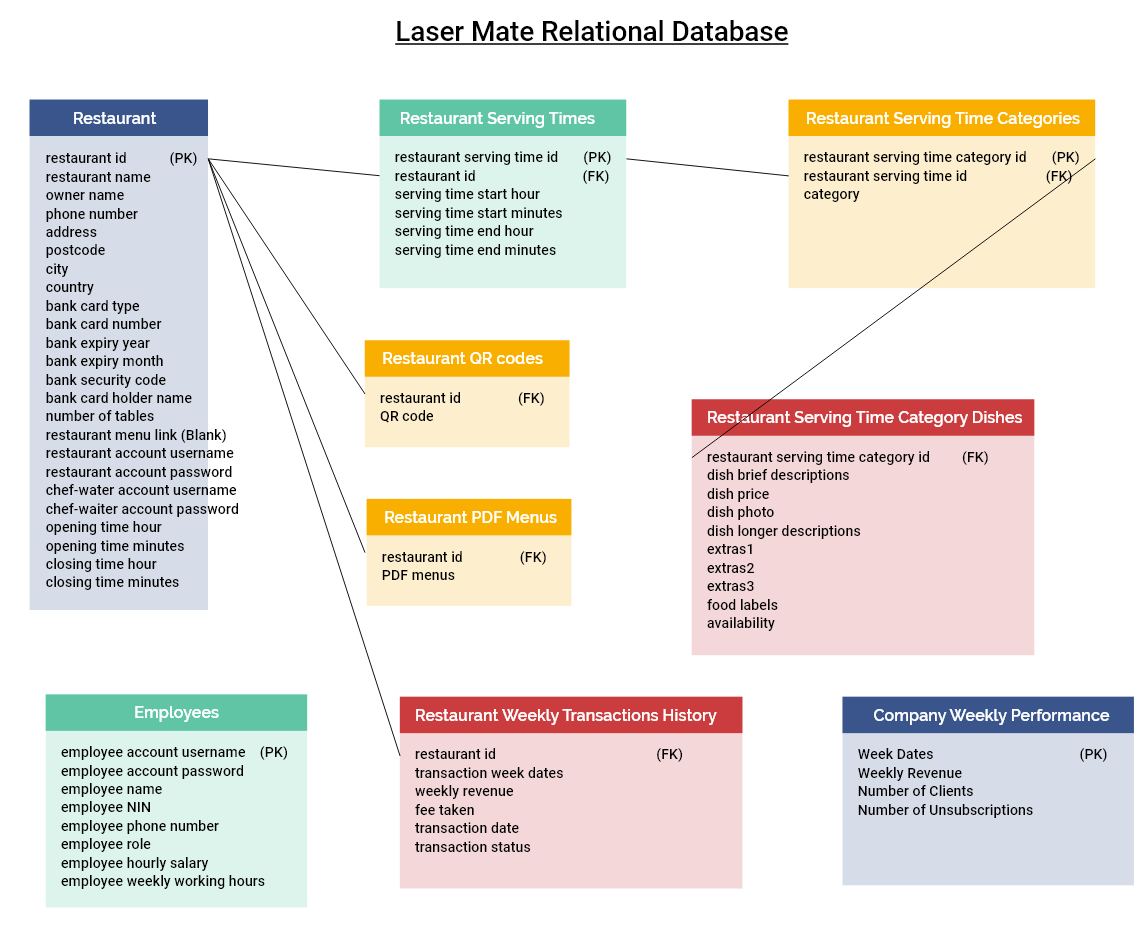
Our platform design also includes animated page interaction, enhancing system’s usability. These include horizontal swiping within the same interface and across different pages, button animation, and timed animation for the “item added” page.

Another essential aspect of digital prototyping is the adoption of the design principles stated above. By applying the aforementioned metrics, hopefully, we have further improved the customer software journey.

You could learn these techniques in YouTube. You could search and filter relevant contents by the channel and the specific topic.

### Database Organization v1s

PostgreSQL is the final database platform option since it could manage intensive volume of user traffics and data processing, such as photos, text, and bank transactions. Examples of similar platforms that use PostgreSQL includes Instagram, Netflix, Trivago, and Figma.



The database design for our software implementation follows the standard relational mapping. Each relation will have a primary/foreign key that uniquely identifies a tuple. Data fields that share a one-to-one relationship under the same data category will place in the same relation.

For data sets that share a one-to-many relationship, we will create a new relation which brings the primary key of the “one relation” as a foreign key. This will allow us to have many tuples based on that unique foreign key. If a relation requires a unique identification for further primary-foreign key mapping, the tuple will create a primary key that uniquely represents the full tuple, including the foreign key of the mapping relation.

Our database design does not contain a many-to-many relationship.

### 1 Overall Coding Strategies current

A key to successful software engineering is to record your programming methodologies, allowing the team to follow consistent strategies that undergo regular evaluation and refinement. The overall workflow is to find and work on a trial version template project that helps you understand the main software development approach.

Type of Recommended Resources

Before discussing the coding aspects, we could first take a deep dive on the types of resources to reach for by comparing the cost-benefits of each coding tutorial platforms. We should prefer video tutorials over text documentation since it is easier when someone explains the codes line by line to you. A drawback of text documentation is that it is harder to understand the contents as you do not have a full visual representation of the programming processes and outcomes. It is easy to forget what is learnt because it is difficult to focus when reading texts for a lengthy period of time without much visual inputs.

LinkedIn Learning could provide effective resources; however, the contents are quite basic and theoretical. They do not have a complete project that attempts to cover all aspects of mobile web development. The other issue is that they do not have a GitHub project example for code referral, making it difficult to understand the project as a whole by reading the relevant codebase. Another similar platform is Udemy, which has a large collection of coding tutorials, however, each tuition costs around £10 and you need to take your chances to pay for the course before viewing it.

The platform I would recommend is YouTube. With strong coaching community, all tutorials are in videos, well sectioned by headings. When having issue with a particular codebase, you could fast forward to the specific tutorial section using the hints within the video progress bar. YouTube presents the largest collection of coding tutorials, tackling the widest range of issues. The codebase would be more up to date to counter for version control within languages and most channels would have GitHub coding examples for future code referral and debugging.

Find, Filter and Prioritize Programming Tutorials

Before searching for relevant video tutorials on YouTube, an essential task is to understand the project programming requirement, giving yourself a few keywords and criteria to search for. The videos must demonstrate work in Django - front-end, and backend - since they are the main programming languages for the project. The tutorial contents should cover data rendering (search, create, update, and delete records) within the front-end, login and registration. It should have a GitHub project template in the YouTube descriptions, enabling you to look back to the codebase when getting stuck with a specific coding part. Ideally, the video should be similar to your project in terms of software features.

Based on these criteria, you should be able to find 3-5 relevant coding examples. By having a wider scope of example solutions for prioritization, you are more likely to find a good example project to work with. The other goal of this filtering process is to help you use other video tutorials if you find the ones you are watching not quite relevant.

Working on the Trial Project

The next phase is to go through the entire tutorials in YouTube and start absorbing the programming methodologies. A reason to aim for watching the full videos is that you want to understand what the YouTuber is teaching you, such as why a particular line is important, the reason they are there, testing strategies and package installation. Without watching the videos, you are skipping the fundamentals of the subjects, making your later programming tasks more challenging. Another point to discuss is whether to watch parts or the full tutorials before starting programming. The advice is first ensuring that most of the tutorials are relevant to your projects, then watching all the essential parts until the tutorials becomes irrelevant. This will help you understand the codes in a bigger picture. You are more likely to make sense of the codes when you do it in a shorter time interval than spacing the work out over a few weeks. To accelerate your programming progress, a good practice is to not hand-type every code snippet while watching the videos. Instead, you should have the GitHub codebase next to you and just copying and pasting the relevant parts into your trial project.

Coding the Real Project

The next phase of the project is coding the platform based on the trial project. My recommendation is to play parts of the video to familiarize again what to do then pause the video to execute that part. Having this coding example and an understanding of the project should enable you to code this particular part without much error. When encountering obstacles, you could replay the video or referring to the specific codebase in the example GitHub. Other sources of information could be helpful, such as Stack Overflow and the web in general. Once you successfully program the software, then continue to work on the next part of the tutorial until you finish the software feature. While working on each part of the software, you could make a note the methodology and resources that are helpful for other colleagues to follow for faster coding progression.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* write test cases while doing the project. It allows you to double check your work if you do test cases while you are doing that part and after you have the full project.

## Evaluation

A crucial aspect of the project is to perform evaluations after the paper prototyping, written up the digital wireframe and developed the software, respectively.

### Evaluation for Paper Prototype

The evaluation process for the paper prototype involves the Jakob Nielsen’s Heuristic Evaluation [1]. Using the 10 standard system usability criteria, we identify and alter the software aspects that could enhance the user experience. You can find the applicable principles in the “Justifying Design Options” section.

<https://www.nngroup.com/articles/ten-usability-heuristics/>

Subsequent to heuristic evaluations, we conduct a literature review to evaluate and integrate the relevant software components into our app design. A brief google search suggests that there are 2 companies from which we consider our designs. They are Dines, and Yo! Sushi. Relevant feature extractions in Dines include the allergy section, and the extras options. Yo! Sushi present an important design suggestion as their way to display the four meals looks extremely appealing.

Semi-Structured Interview & Questionnaire

* Supervisor
* Family member

### Evaluation for Digital Prototype

* Paste your google form here

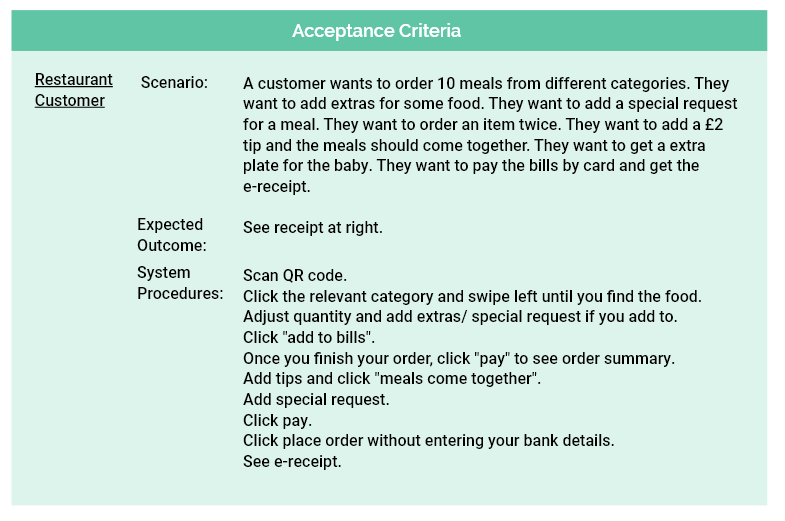
Cognitive Walkthrough & Semi-Structured Interview & Questionnaire

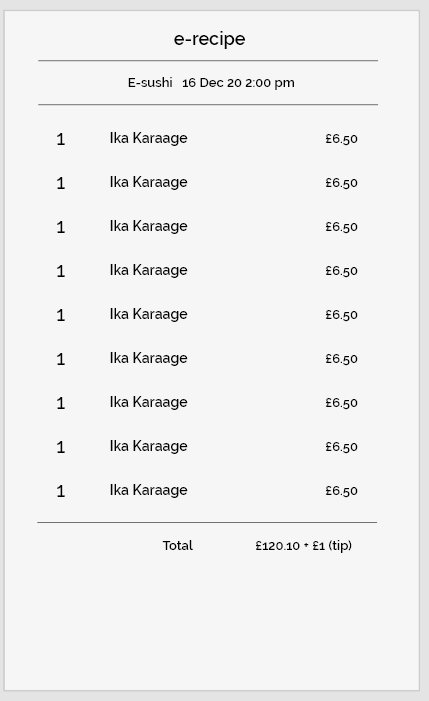
* Supervisor
* Family member
* Friends

### Testing Documentations

* Paste your expected and actual outcomes here
* Postman – teach the reader how you use it for API testing

Acceptance criteria describes an example of practical tasks the system could perform. As this scenario will feed into software testing, it will also include an expected outcome which compares with the system’s actual outcome. System procedures is another aspect of the acceptance criteria that describes the instructions that implements the aforementioned scenario.





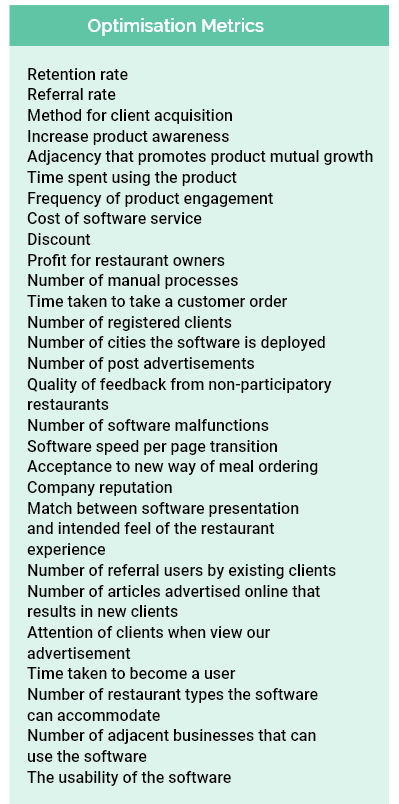
### Evaluation for Software Product

Cognitive Walkthrough & Semi-Structured Interview & Questionnaire

* Supervisor
* Family member
* Friends

### Future Work v1

The company should identify future work to outperform existing practices. By understanding the optimization metrics that determine the product’s success, it would make it harder for other ventures to build a better product. The critical aspects for consideration include keeping customers continue using the system and will not leave; how we can grow the business internationally; and how to maintain system capability. Furthermore, we could consider the points mentioned in the “Value Proposition Canvas” and apply some relevant growth strategies.



# Bibliography

# Appendix

# Video Presentation

1. Demonstrate the final product & the test cases using Q.R. code videos and screenshot photos

== not in dissertation but video=== testing

Customer

* Scan QR code
* workflow to order 10 meals from all different categories; add special requests; add extras; increase meal quantity.
* Read order summary; add quantity; check meal descriptions and prices; swipe through meal over a few pages.
* Check total price; give tips; check new price;
* Change meal together and meal separately;
* Add special requests.
* Order and produce e-receipt

Restaurant Owner

* Account registration; email account confirmation; see whether new account is automatically aggregated in the company staff account
* Forgetting password; email password retrieval;
* Add serving time; add categories; add meal details; edit serving time to check whether other data (categories, meal details) will be changed
* Delete serving time; check whether other data (categories, meal details) will be deleted
* Add restaurant info data; see whether restaurant name and address will be changed in the customer interface; check whether restaurant phone number and owner name will be changed in the company staff interface
* Check whether business info Q.R. code works; whether it shows the customer interface for the restaurant.
* Check whether help page will redirect user to video documentation page.

Company Staff

* do later \*\*\*

=============